Stress Level Detection After Sleep Using Supervised Machine Learning

Sophia Vellozzi
Department of Computer & Information Science & Engineering
University of Florida
Gainesville, FL
s.vellozzi@ufl.edu

***Abstract*—Research suggests that there is a strong relationship between a person's stress level and their sleep [1]. This study provides a comparison of several supervised classification machine learning algorithms (logistic regression, Naïve Bayes, random forest, K-Nearest Neighbors or KNN, support vector machine or SVM, and AdaBoost) to predict the stress levels of a person based on various sleep factors captured in the Smart-Yoga Pillow (SaYoPillow) dataset [2]. This dataset presents a novel approach to measuring sleep factors including snoring rate, respiration rate, body temperature, sleeping hours, heart rate, limb movement, blood oxygen, and eye movement. Results show that the Naïve Bayes, random forest, KNN, and SVM performed the best (accuracies of 1.0), followed by logistic regression (accuracy of 0.972), and then AdaBoost (accuracy of 0.956) in predicting stress levels following sleep.**

Keywords*—Classification, data preprocessing, exploratory data analysis, hyperparameter tuning, model evaluation, supervised machine learning

I. Introduction

Machine learning, a transformative subset of artificial intelligence, has emerged as a powerful tool for extracting patterns and insights from complex datasets. Traditionally employed in diverse fields such as image recognition, natural language processing, and recommendation systems, machine learning is now being leveraged to unravel the intricate relationship between personal well-being and daily habits.

In this study, we delve into the application of supervised machine learning to investigate the correlation between an individual's sleep patterns and their stress levels. Supervised machine learning is a type of machine learning where an algorithm learns a mapping between the input data and corresponding output labels based on a labeled training dataset. In this framework, the algorithm is provided with pairs of input and output examples, allowing it to generalize and make predictions on unseen data. Classification algorithms, a common type of supervised learning, are designed to categorize input data into predefined classes or labels. These algorithms aim to learn decision boundaries that separate different classes, enabling accurate assignment of labels to new, unseen instances based on the patterns learned during training. This paper presents a performance analysis from implementing various classification algorithms to understand the relationship between a person's sleep and their stress levels and seeks to compare the models to understand which performs best on the data and why. Several machine learning concepts and frameworks were used with the data and models, including exploratory data analysis, train-test split, feature scaling, Principal Component Analysis (PCA), and hyperparameter tuning algorithms.

The application of machine learning to studying a person's sleep helps offers a more nuanced approach to understanding the factors that influence a person's mental and physical well-being even beyond just sleep and stress. Thus, the analysis of several machine learning algorithms and their performance is essential in this space to gain better healthcare insights and contribute to scientific advancements and public health.

II. Methods

A. Data

1) Background: The dataset used in this analysis is the "Human Stress Detection in and Through Sleep" data on Kaggle [2]. It includes data collected from the Smart-Yoga Pillow (SaYoPillow), which is a smart device that tracks various factors during sleep. The data has 630 rows and 9 columns. This dataset includes the relationship between 8 factors (snoring rate, respiration rate, body temperature, limb movement rate, blood oxygen levels, eye movement, number of hours of sleep, heart rate) and stress level. The stress levels are given numbers from 0 to 4, where 0 indicates low-normal stress, 1 indicates medium-low stress, 2 indicates medium stress, 3 indicates medium-high stress, 4 indicates a high stress level. The 8 factors are the input features for our models and the stress level is the output or label.

2) Exploratory Data Analysis: The data was loaded into Jupyter Notebook as a CSV file to create a dataframe using Python's Pandas library before beginning data preprocessing and the exploratory data analysis.

In conducting exploratory data analysis (EDA) on the dataset, a comprehensive examination of key features pertaining to sleep patterns and stress levels was undertaken to lay the groundwork for subsequent modeling. Descriptive statistics, graphical representations, and correlation analyses were employed to gain insights into the distributions and potential relationships within the data. Visualization techniques, including histograms and scatter plots, provided a nuanced understanding of the distributional characteristics and potential associations between variables. Specifically, the scatterplots that plotted each of the input features against the stress level revealed a linear relationship between the input and output variables.

The correlation coefficent, or the strength of the linear relationship between the variables, was computed using the Pandas corr() method. As shown in Table I, all features had strong linear correlations with the output. The features snoring_rate, limb_movement, heart_rate, respiration_rate, and eye_movement showed strong positive linear correlations, and blood_oxygen, body_temperature, and sleeping_hours showed strong negative linear correlations.

This EDA phase not only facilitated a deeper understanding of the dataset's characteristics but also informed decisions regarding feature selection for subsequent machine learning modeling.

TABLE I. CORRELATION COEFFICIENTS

Feature	Correlation Coefficients
snoring_rate	0.975322
limb_movement	0.971071
heart_rate	0.963516
respiration_rate	0.963516
eye_movement	0.951988
blood_oxygen	-0.961092
body_temperature	-0.962354
sleeping_hours	-0.973036

3) Train-Test Split: The data was split so that 80% went into the train set and 20% in the test set. The resulting shapes of the train and test sets are as follows: X_train shape = (504, 8), y_train shape = (504,), X_test shape = (126, 8), y_test shape = (126,).

4) Feature Scaling: The features had varying scales of data, so feature scaling was implemented to ensure each feature contributes proportionally in the training process. MinMaxScaler was used to transform the features to a consistent scale between 0 and 1 without changing the current shape of the original data distribution.

5) Principal Component Analysis (PCA): Lastly, PCA was used to reduce the dimensionality of the data since there were a large number of features. By selecting a reduced number of principal components, the curse of dimensionality is alleviated, leading to more computationally efficient model training. Additionally, PCA aids in simplifying the interpretation of the data by focusing on the most informative components. This reduction in dimensionality not only accelerates the modeling process but also helps in identifying and retaining the most significant patterns in the dataset. Two components were used for the PCA, denoted as PC1 and PC2.

B. Models

In this study, a diverse set of classification algorithms was selected to comprehensively analyze the intricate relationship between sleep factors and stress levels. The chosen algorithms were logistic regression, Naive Bayes, random forest, K-Nearest Neighbors (KNN), support vector machines (SVM), and AdaBoost. The rationale behind this selection stems from the unique strengths and characteristics of each algorithm. Logistic regression and Naive Bayes were chosen for their simplicity, interpretability, and efficiency in handling classification tasks. Random forest, an ensemble method of decision trees, was selected for its ability to capture complex relationships and provided a way to compare ensemble learning techniques to simple machine learning models. K-Nearest Neighbors provides a distance-based approach, offering insights into local patterns within the data. Support vector machines help in capturing intricate decision boundaries in high-dimensional spaces, making them apt for nuanced relationships. Finally, AdaBoost was included to harness the power of boosting and compare ensemble methods with traditional machine learning approaches to see which improves the overall predictive performance.

As mentioned previously, the dataset was preprocessed, including feature scaling using MinMaxScaler and dimensionality reduction through Principal Component Analysis (PCA) to enhance computational efficiency before applying the algorithms.

Subsequently, each classification algorithm was trained on the preprocessed dataset, using appropriate hyperparameter tuning when necessary. The hyperparameter tuning algorithm GridSearchCV was used to find the best combination of hyperparameters for the random forest and KNN models, and RandomizedSearchCV was used for the AdaBoost model.

Cross-validation was performed using 10 randomized folds for each of the models. Following cross-validation, the final models were fitted on the training set and then evaluated on the separate test dataset, allowing for an unbiased assessment of their predictive accuracy and generalization to unseen data.

Lastly, the models were evaluated using common classification metrics such as accuracy, precision, recall, and F1-score, and these results were displayed in confusion matrices and classification reports. This methodological approach ensures a robust and comparative analysis of diverse algorithms, capturing the complexity of the sleep-stress relationship through a multifaceted lens.

III. RESULTS

Below are summaries of the accuracy, precision, recall, and F1-score for each of the models. Accuracy measures the overall correctness of the model by calculating the ratio of correctly predicted instances to the total instances. Precision is a measure of the accuracy of positive predictions, representing the ratio of correctly predicted positive observations to the total predicted positives. Recall is a measure of the model's ability to capture all relevant instances, determined by the ratio of correctly predicted positive observations to the total actual positives. F1-score is the harmonic mean of precision and recall, offering a balanced assessment, especially in situations with uneven class distribution. The formula for F1-score is represented as 2 * ((Precision * Recall) / (Precision + Recall)).

1) Logistic Regression: The logistic regression model used hyperparameters of `max_iter = 1000` and `C = 0.1`. The cross-validation accuracy was ~0.972. The accuracy after fitting the model was ~0.992. Table II. shows the classification report, and Fig. 1 shows the confusion matrix.

TABLE II. LOGISTIC REGRESSION CLASSIFICATION REPORT

	precision	recall	f1-score	support
0	1.00	1.00	1.00	23
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	28
3	1.00	0.96	0.98	26
4	0.96	1.00	0.98	25
accuracy			0.99	126
macro avg	0.99	0.99	0.99	126
weighted avg	0.99	0.99	0.99	126

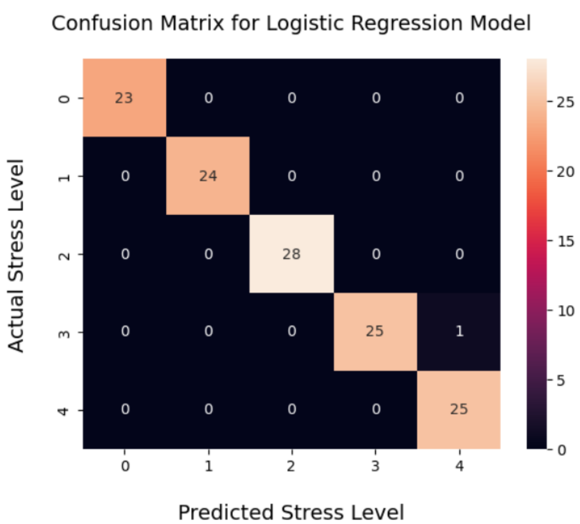

Fig. 1. Confusion matrix for logistic regression

2) Naïve Bayes: The cross-validation accuracy was 1.0. The accuracy after fitting the model was 1.0. Table III. shows the classification report, and Fig. 2 shows the confusion matrix.

TABLE III. NAÏVE BAYES CLASSIFICATION REPORT

	precision	recall	f1-score	support
0	1.00	1.00	1.00	23
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	28
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	25
accuracy			1.00	126
macro avg	1.00	1.00	1.00	126
weighted avg	1.00	1.00	1.00	126

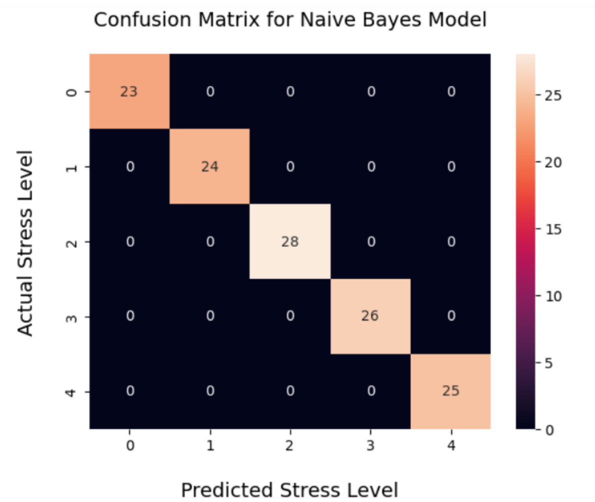

Fig. 2. Confusion matrix for Naïve Bayes

3) Random Forest: The hyperparameters used as a result of running the GridSearchCV algorithm were `max_depth = None`, `min_samples_leaf = 1`, `min_samples_split = 2`, and `n_estimators = 100`. The cross-validation accuracy was 1.0. The accuracy after fitting the model was 1.0. Table IV. shows the classification report, and Fig. 3 shows the confusion matrix.

TABLE IV. RANDOM FOREST CLASSIFICATION REPORT

	precision	recall	f1-score	support
0	1.00	1.00	1.00	23
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	28
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	25
accuracy			1.00	126
macro avg	1.00	1.00	1.00	126
weighted avg	1.00	1.00	1.00	126

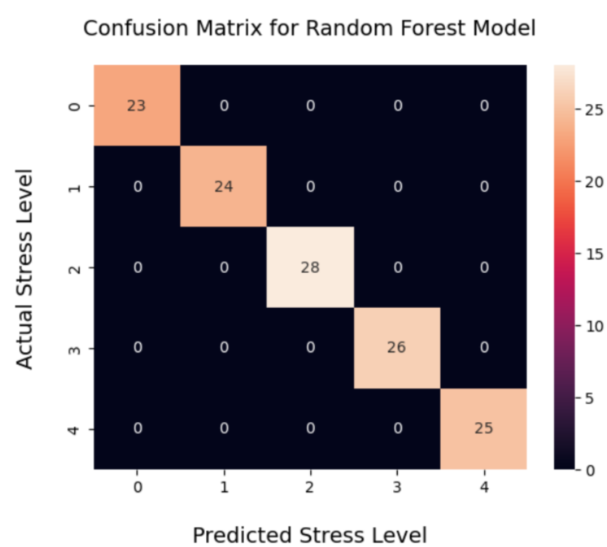

Fig. 3. Confusion matrix for random forest

4) K-Nearest Neighbors (KNN): The hyperparameters used after applying the GridSearchCV algorithms were `leaf_size = 15`, `n_neighbors = 3`, and `weights = uniform`. The cross-validation accuracy was 1.0. The accuracy after fitting the model was 1.0. Table V. shows the classification report, and Fig. 4 shows the confusion matrix.

TABLE V. KNN CLASSIFICATION REPORT

	precision	recall	f1-score	support
0	1.00	1.00	1.00	23
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	28
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	25
accuracy			1.00	126
macro avg	1.00	1.00	1.00	126
weighted avg	1.00	1.00	1.00	126

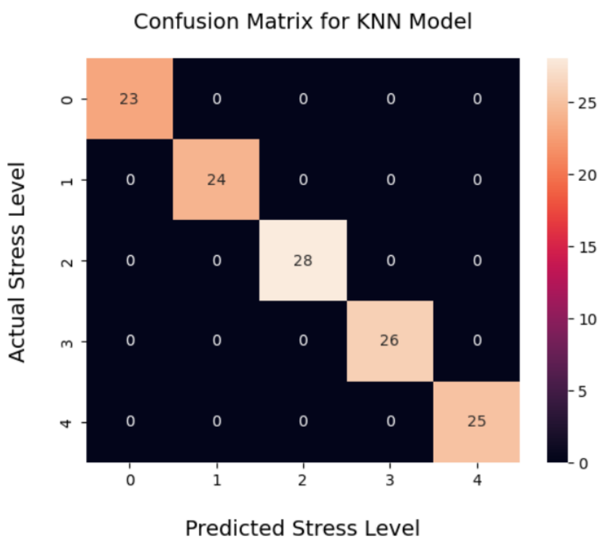

Fig. 4. Confusion matrix for KNN

5) Support vector machine (SVM): The hyperparameters used in training the model were `max_iter = 1000` and `kernel = 'rbf'`. The cross-validation accuracy was 1.0. The accuracy after fitting the model was 1.0. Table VI. shows the classification report, and Fig. 5 shows the confusion matrix.

TABLE VI. SVM CLASSIFICATION REPORT

	precision	recall	f1-score	support
0	1.00	1.00	1.00	23
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	28
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	25
accuracy			1.00	126
macro avg	1.00	1.00	1.00	126
weighted avg	1.00	1.00	1.00	126

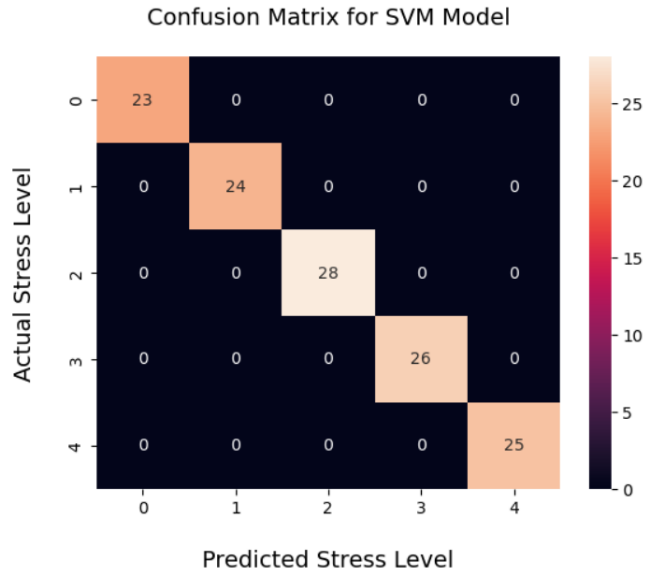

Fig. 5. Confusion matrix for SVM

6) AdaBoost: The hyperparameters used after applying the RandomizedSearchCV algorithm were `estimator = DecisionTreeClassifier(max_depth=1)`, `learning_rate = 0.2567367263121657`, and `n_estimators = 76`. The cross-validation accuracy was 0.956. The accuracy after fitting the model was 1.0. Table VII. shows the classification report, and Fig. 6. shows the confusion matrix.

TABLE VII. ADABOOST CLASSIFICATION REPORT

	precision	recall	f1-score	support
0	1.00	1.00	1.00	23
1	1.00	1.00	1.00	24
2	1.00	1.00	1.00	28
3	1.00	1.00	1.00	26
4	1.00	1.00	1.00	25
accuracy			1.00	126
macro avg	1.00	1.00	1.00	126
weighted avg	1.00	1.00	1.00	126

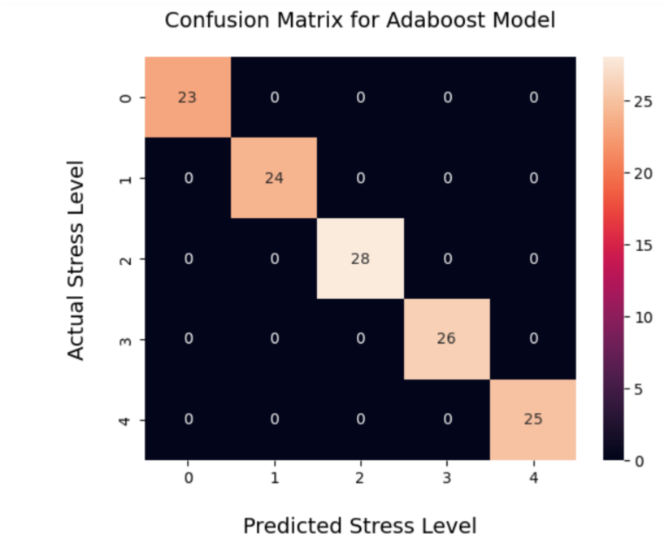

Fig. 6. Confusion matrix for AdaBoost

IV. DISCUSSION

The obtained results from the classification algorithms shed light on the effectiveness of each model in discerning the intricate relationship between sleep patterns and stress levels. All the models demonstrated great performance and impressive accuracy for both cross-validation and on the test set. As a result, the models did not overfit or underfit. The Naïve Bayes algorithm exhibited exceptional accuracy, achieving a perfect score of 100% both in cross-validation and on the test set. The simplicity of Naïve Bayes proved advantageous in capturing the inherent structure within the dataset. Similarly, the Random Forest, K-Nearest Neighbors (KNN), and Support Vector Machine (SVM) models each attained a perfect accuracy of 100% on both cross-validation and the test set. The effectiveness of these models, even after hyperparameter tuning, underscores their ability to grasp complex relationships within the data.

The confusion matrices provided visual insights into the models' classification performances, revealing a high degree of correct predictions across all models. The precision, recall, and F1-score metrics, detailed in the respective classification reports, further affirm the models' capabilities in correctly classifying instances related to sleep and stress. Each of the models achieved perfect or close to perfect scores across all metrics, reflecting their robust performance.

The hyperparameter tuning process, especially evident in Random Forest, KNN, and AdaBoost models, played a pivotal role in optimizing the algorithms for the specific nuances of the dataset. Particularly, for the AdaBoost model, the Principal Component Analysis (PCA) helped boost performance from an accuracy of ~85% to ~96% for the cross-validation. Interestingly, the cross-validation accuracy for AdaBoost was less (0.96) compared to some of the simpler models such as Naïve Bayes, KNN, and SVM (1.0). With AdaBoost being an ensemble model and more complex than the others, it could have been more sensitive to the specific subsets of data used in the folds of the cross-validation process. The composition of the training folds could have impacted the model's learning process. Overall, as shown in Fig. 7, Naïve Bayes, random forest, KNN, and SVM showed the highest accuracy (1.0), followed by logistic regression (0.972), and then AdaBoost (0.956).

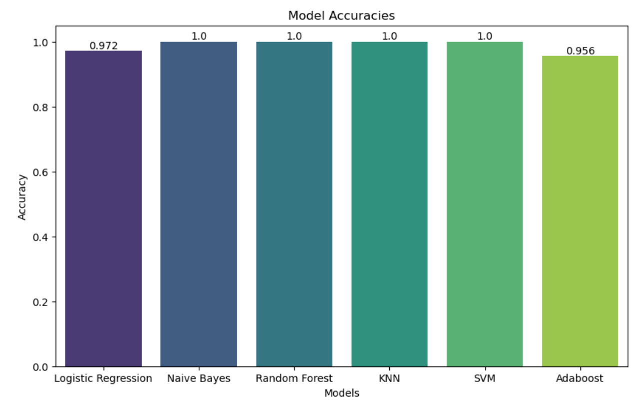

Fig. 7. Bar plot displaying the cross-validation accuracies of each of the models.

V. CONCLUSION

In summary, the comprehensive evaluation of each classification algorithms highlights the successful application of machine learning techniques in understanding the intricate interplay between sleep patterns and stress levels. The exceptional accuracy scores and consistent metric performances underscore the potential of these models for informing personalized interventions and contributing valuable insights to the fields of healthcare and well-being. Moreover, the nuanced exploration of hyperparameters demonstrates the importance of fine-tuning models to extract optimal predictive capabilities, emphasizing the significance of machine learning methodologies in unraveling complex relationships within multidimensional datasets. While more work should be done with expanding this data to new and different models, this study illustrates how common machine learning techniques have great potential in predicting the relationship between sleep and stress.

REFERENCES

[1] L. Rachakonda, S. P. Mohanty, E. Kougianos, K. Karunakaran, and M. Ganapathiraju, "Smart-Pillow: An IoT based Device for Stress Detection Considering Sleeping Habits", in Proceedings of the 4th IEEE International Symposium on Smart Electronic Systems (iSES), 2018, pp. 161--166.

[2] L. Rachakonda, A. K. Bapatla, S. P. Mohanty, and E. Kougianos, "SaYoPillow: Blockchain-Integrated Privacy-Assured IoMT Framework for Stress Management Considering Sleeping Habits", IEEE Transactions on Consumer Electronics (TCE), Vol. 67, No. 1, Feb 2021, pp. 20-29.

[3] A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras, and Tensorflow Concepts, Tools, and Techniques to Build Intelligent Systems*. Beijing: O'Reilly, 2023.